Paper 364-2011

What Happens When You Push the Easy Button®? - Migrating to BI 9.2

Faron Kincheloe, Baylor University, Waco, TX

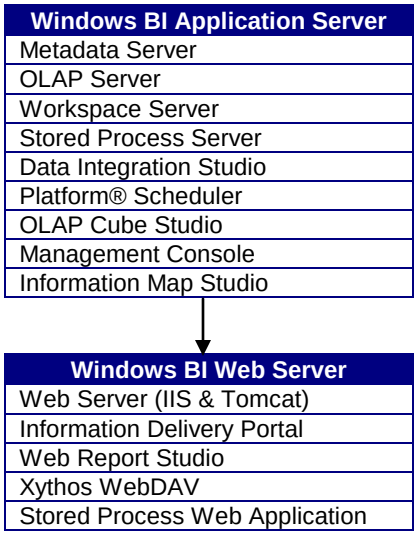
ABSTRACT

It has been said that the utility for migrating a Business Intelligence (BI) installation from version 9.1 to version 9.2 is supposed to be like pushing the "Easy Button®". Baylor University has successfully completed a migration of Enterprise Business Intelligence to version 9.2. This paper will describe the migration experience and present some lessons learned in the process.

INTRODUCTION

Baylor University's BI installation was three years old. It was a full implementation of SAS® Enterprise BI Server and SAS® Data Integration Server. The primary purpose of this installation was to deliver information for administrative decision making to over 250 registered users across campus without having to install any software on the client computers. All users log into the Information Delivery Portal where the content is distributed in a number of formats depending upon the user's membership in one or more BI security groups. Content formats include: static PDF, HTML, or Excel files, most of which are updated periodically by scheduled SAS® jobs, stored processes with custom forms that allow the user to execute the process based on selected parameters, and Web Report Studio (WRS) sample reports based on tables and cubes. These WRS samples provide a starting point from which users can change fields and filtering to suit their needs and save their customized reports for future use. Enterprise Guide and the Add-In for Microsoft Office had not been made available to the user community. There are no additional "SAS® Solution" products installed at our site.

Two new Windows servers were purchased at the time of the original installation to house the BI environment. These servers still had two years of life remaining according to University policy. One server houses the web components. The other houses all of the other BI components. Apache Tomcat was chosen as the web application server software because there was no cost for licensing and we had experience with this software from a previous data warehouse. The diagram below provides an illustration of the BI environment.



The BI Platform has been administered solely by three staff members in Baylor's Office of Institutional Research and Testing. These three were also responsible for the migration while continuing to perform their regular duties in the office.

THE MIGRATION PROCESS

The Baylor IT department had moved the University standard web browsers to versions that were not supported by our current version of BI. There were some nagging issues with BI and the underlying SAS® version 9.1.3 that it did not appear were going to be fixed in that version. It was apparent that all development efforts were going to version

9.2 and we were hearing about new features and improvements that we would like to have. Our installation package was created before BI Dashboards were packaged with the Information Delivery Portal and we had been unable to get the BI Dashboard software for installation. With two years of life still left on the servers we did not want to wait to upgrade the BI Platform in conjunction with the hardware replacement. All of these were factors that motivated us to perform the migration when we did.

The migration project began around the first of December 2009 and culminated with the actual migration to production on July 17, 2010. This length of time does not accurately reflect the actual amount of time required as there were a number of factors not directly related to the migration that affected our completion date.

The following list from the Migration Guide summarizes the steps required to install SAS® 9.2 and migrate SAS® 9.1.3 content:

1. Design your migration.
2. Perform pre-migration tasks.
3. Install SAS® 9.2 and migrate your SAS® 9.1.3 content.
4. Perform post-migration tasks.
5. Validate your migration.

Designing your migration means reviewing the SAS® 9.2 requirements (hardware, software, and migration) against the current SAS® 9.1.3 deployment, and developing a plan for how to get the SAS® 9.1.3 content-- data and configuration--integrated into a SAS® 9.2 system. In one presentation on migration at a SAS conference, the presenter said the preferred method was to have different hardware as the destination of the migration. As mentioned earlier, our servers still had two years of useful life and there were no other servers available for the migration. Therefore, we had no choice but to do an “in place” migration.

One of the choices that must be made during the design phase is whether to do migration, promotion, or a combination of the two. Our scenario met all the criteria for a migration. Promotion would require separate installation and configuration of the BI Platform. It was also our understanding that we would not be able to promote any of our users’ personal pages in the portal. With these factors in mind, migration alone was determined to be the best course of action for us.

Since we would be performing the migration on the same hardware as our existing BI environment, we were concerned about the risk of damaging the production installation while testing the migration. Even though we had no desire to maintain parallel installations once the migration was completed, we could not afford to have any significant down time during the migration process. We were not confident in our ability to create a new instance without port conflicts. We also wanted to minimize the confusion as to which system a person would be logged into at any given time. At the beginning of this project we only had a production BI environment. There was no development environment. Therefore, we decided to create a clone of our version 9.1.3 BI Platform so we could do a test migration first. Fortunately, the Baylor IT department has a virtual server farm. They were able to create two virtual servers to correspond to the two production servers that house our BI installation. We did everything we could to replicate our production environment as closely as possible. The directory structure was identical. Even the server names were within one letter of being the same. The only limitation was that they were not able to allocate as many resources in terms of CPU, memory, and disk space. At one point there was some concern that we did not have enough resources to run the BI Platform but we managed to get by accepting the fact that system response would be slower than the production environment. As far as software installation and basic configuration is concerned, it was much more difficult to create the 9.1.3 clone than it was to migrate to 9.2.

We finished the cloning process on January 25, 2010, almost 2 months into the project. This time included more than a week of holidays and time spent on other duties in the office. Then we ran the migration analysis and spent some time figuring out how to best respond to the items identified in the analysis reports. We completed the automated portion of the migration on February 24, 2010. The automated migration had to be restarted several times because we ran out of disk space on the virtual server where we were trying to install the web applications. At this point we had a fully functional BI installation. However, it was not yet usable due to a number of security modifications and other manual configurations that were required for the new version. These security modifications were identified by a security analysis report that can be run once the automated migration to 9.2 is completed. This analysis also took some time to digest in order to fully understand the impact of the new version. By April 8, 2010 we had completed all of the manual post-migration tasks and were in the process of testing the new installation.

Our test procedure was to perform all functions that we would normally perform in the use and administration of the BI Platform. We placed a test user in each of the security groups one at a time. Then we logged into the portal and verified that the user had access to the content and functionality for that group. We also performed searches to ensure the user could not access unauthorized content. We added and removed users. We tested our unscheduled load processes in SAS® Data Integration Studio. We triggered all of our scheduled jobs in Flow Manager to ensure that they would run properly. Our testing was complete by the end of April. At this point we had identified about a half dozen items that we still needed to investigate. Most of these were related to our own processes and customizations and would not have prevented us from going live with the migration. However, we decided it would be best from a timing and workload standpoint to wait until the end of spring semester and summer enrollment processing was complete. All of our outstanding items had been resolved by June 23 and Saturday, July 17, 2010 was set as the production migration date.

We spent about an hour on Friday afternoon before migration day to install the JDK on both servers and put a basic JBoss installation on the web server. We stopped the Tomcat service Friday evening so no one could log on to the portal. We began the actual migration around 7:30 Saturday morning after verifying that scheduled jobs from the night before had completed successfully. Since there were not a lot of tasks that could be performed simultaneously in the beginning, all three team members worked together on the automated portion. One person read the procedure, a second person entered the commands on the servers and the third person proofread what was being entered. This may seem like overkill but typos can be difficult to locate and correct when they affect metadata or parameters buried in obscure configuration files. This step was finished by 11:30 a.m. The 9.2 software had been installed and all metadata and web content had been migrated to the new version. Since we were doing an in place migration, there was no need to relocate external datasets or programs. At this point we divided up the remaining manual configuration tasks so that each team member could stay busy working on separate tasks. When we installed the original BI environment, we realized that the folder structure in Web Report Studio was not intuitive. We decided that migration would be the best time to clean up and reorganize this structure so this was included in our tasks for the day. By 7:30 that evening, after 12 hours, all of the migration, reorganization, and testing steps were completed and the system was ready for production use.

The following paragraphs describe some of the observations and lessons learned by the team in the process of migrating the BI Platform to the new version. It should be noted that this migration was to Maintenance Release 2 of SAS® 9.2. Maintenance Release 3 became available just as we were concluding the testing phase. However, we had already made a commitment to the production migration date and did not want to start over and risk delaying the migration to production. There was a note with the maintenance release announcement stating that several SAS® products will have new releases that depend on the third maintenance release for SAS® 9.2. It also suggested that we might want to wait until the new product release before applying the third maintenance release. We decided it would be more efficient to wait to test and install a wider range of products at one time. Some of the issues we experienced may have been resolved in the third maintenance release, but for the most part our lessons and observations should be applicable regardless of which maintenance release is being installed.

IT WAS A BLESSING TO BE PATCHED

Another SAS customer indicated they had trouble figuring out which hot fixes they needed to apply in order to be at the correct level for migration. Fortunately, we did not have this problem. When we originally installed the BI Platform we made the decision to apply relevant patches on a regular basis. As new hot fixes were released, we would download them to a folder as part of our software depot. About once a month we would take the system down for about an hour to install hot fixes, operating system updates, and any other software updates that were available. The administrator responsible for this did a good job keeping a journal listing of all the hot fixes and the date they were installed. We were current when it came time to do the migration and it was easy to make sure our clone installation was current as well. All we had to do was access the hot fix depot and install them in the order listed in the journal. The SAS Migration Utility (SMU) is supposed to identify patches that need to be installed. Since we were up to date, we were surprised to see a message in the SMU analysis report that hot fix E9BC13 needs to be applied. Apparently, the SMU cannot determine the presence of this hot fix so it issues this standard warning as a reminder to verify that the hot fix has been installed.

PAY CLOSE ATTENTION TO THE MIGRATION ANALYSIS REPORT

We sought to understand and respond appropriately to all of the messages in the migration analysis reports. After the migration we found out there is documentation for all possible messages at the SAS website http://support.sas.com/rnd/migration/utility/messages.html. One of the messages in our analysis was, “NOTICE: there are (190) "unmatched" home folders, these are home folders not belonging to a SAS Metadata user (probably "Public users"). They will be migrated "in place", and not relocated to a new SAS 9.2 Home Folder.” The wording in this message made it seem innocuous. I feel certain that we brought it up when we were discussing the reports with technical support. However, no attention or concern was given to this message and since we did not understand its significance the message was ignored. When we were verifying our test migration, we observed that all of our users’

personal web reports had been migrated but were buried deep in the metadata folder structure instead of being in the "My Folder" location for each user. We just assumed that this was a normal result of the migration. It was not until we were finishing the production migration that we realized the connection between the analysis message and the location of the web reports. We have since found out that this is one of the shortcomings of the migration process that cannot be prevented. There is supposed to be a way to manually relocate the reports to the appropriate location. However, we were unable to manually move the reports for our users so they will have to relocate the reports to their personal folders themselves. The lesson we learned is that we should investigate further anything that seems abnormal and not dismiss messages so readily. If we had investigated further, perhaps we would have been able to find a way to move the reports prior to the production migration. At least we would have had a better understanding of what was happening.

PRACTICE MAKES PERFECT

Since we were doing an "in place" migration, it was intended that the migration would mark the end of the version 9.1.3 BI Platform. We would retain backups so that we could revert back if necessary but this was not an acceptable option. We wanted to keep down time to a minimum and did not want any surprises during our final migration. For this reason, we invested the time and effort necessary to clone our BI installation and do a practice migration first. One of the goals of the practice migration was to produce a "script" that we could follow to ensure that the production migration did not deviate in any way from what had been tested. The migration utility has the capability of recording a response file that can be replayed to repeat a migration. However, we wanted to be able to repeat every manual step as well so we chose to create a manual script that documented all of our responses, manual configurations and customizations. We took copious notes of everything we did. If we worked with technical support we kept a journal of any changes that were made. We supplemented this by copying relevant instructions out of the migration manual and from white papers and technical notes that we were provided by SAS. What we ended up with was a 42 page document describing, in detail, every step that was required to migrate our specific BI environment. In preparation for the final migration, we prepared an assignment sheet or checklist. This sheet identified the specific step, the pages in our document that described the step, the person responsible for this step and any steps that must be completed prior to this step. There was also a space to sign off completion of the step. Each person was given a copy of the checklist and copies of pages from the document for which they were responsible. The master copy of the checklist was centrally located in the office so that, at any time, we could check to see if we could proceed to our next task or if we needed to wait on someone else to finish a task. A copy of the checklist has been provided as an appendix to this paper. This process went almost flawlessly except for one delay caused by parameters that were copied directly from a technical note instead of from the actual configuration file where changes were made. More details of this will be provided later in this paper.

BACKUP EARLY AND OFTEN

It is better to have a backup and not need it than need a backup and not have it. We typically think of a backup as something that is put in place after everything else is completed but we found that it can be a great time saver to make backups of the metadata periodically throughout the migration process. The best way to do this is to stop the metadata server and use the operating system to copy the Lev1 folder. The first backup should be made as soon as the migration utility finishes running successfully on the metadata server tier. If you are migrating in a multi-tier environment a backup should be made after each tier is migrated. This will allow you a way to quickly return to your last successful milestone without having to start all over. All you have to do to restore is to stop the metadata server and rename your Lev1 backup folder to Lev1. During the test migration, the mid-tier migration crashed due to issues with disk space. We could not restart the mid-tier migration without undoing changes that had been made to metadata prior to the crash. The first time this happened, the only way to undo the changes was to start the migration over from the beginning. Then we made a backup copy of the Lev1 folder and on subsequent attempts we were able to restart mid-tier migration by simply reverting back to our copy of Lev1. Another good time to make a backup copy is before making any significant changes to security or the system configuration. This makes it easy to start over if things do not go as expected.

SECURITY IS STILL THE BIGGEST SPEED BUMP

When we initially implemented the BI Platform we spent more time and effort on security than on any other aspect of BI. The same was true for our migration to version 9.2. Our security strategy was still valid but to some extent we had to re-implement the strategy due to the security changes in the new version. We spent a considerable amount of time trying to make sense of the differences identified in the security analysis reports and determine how we needed to respond. In addition to the changes identified in the security analysis reports, we stumbled across a few other issues that are discussed below.

Security is no longer inherited through the library.

In version 9.1, metadata security applied to a library definition was inherited by the tables in that library. This is not the case in version 9.2. Libraries and tables now inherit their security from the metadata folder in which they reside. This change caused us more work than any other issue. In 9.1, the default location of a library within the metadata

structure was different depending on whether the library was created in Management Console or Data Integration Studio. We had several libraries and underlying tables that had to be moved into folders to be consistent with the new structure. Security settings had to be moved from all of our libraries up to the parent folder.

Web Report Studio can navigate the entire folder structure.
Previously, Web Report Studio content could only be stored underneath a designated metadata folder. Since our department had the only BI clients that could navigate other folders, the security of these folders was not a concern. In version 9.2, Web Report Studio has the capability of navigating through the entire metadata folder structure unless restricted by security settings. Users started saving their reports in folders reserved for data tables and stored processes. As a result, we had to apply additional security to many of our folders to prevent the user groups from saving reports there while still allowing the groups to access their data and stored processes.

Portal Administrators defined through Access Control Template.
A user now becomes a portal administrator by being granted ReadMetadata and WriteMetadata permissions in the Permission Pattern of a new template called Portal ACT. Previously, portal administrators were users who were members of the Portal Admins group. Perhaps we missed a step somewhere but we discovered the hard way that membership in the group is still necessary to ensure that ReadMetadata permissions are applied throughout the portal content. Shortly after the final migration we had some personnel changes in our department that required us to add a new user to the Portal ACT. Then we received a call that one of our user groups had lost a stored process from their portal page. We put the stored process back only to find it missing again the next day. It was being deleted by a cleanup process each time our new portal administrator logged onto the portal. She had not been added to the Portal Admins group and consequently did not have ReadMetadata access to this particular stored process. Therefore, when she logged on the portal saw this as orphaned content and removed it from the portlet. It is important to verify that portal administrators have access to all the necessary content folders after they are added to the Portal ACT.

User credentials are cached in BI client software if the profile has a saved password.
It was annoying to be asked repeatedly for a user name and password in the older versions of BI clients such as Management Console and Data Integration Studio. It is generally a welcomed convenience that the new version of these clients caches the user credentials when the password is stored in the profile used to log on. However, this new feature caused some problems for us when we were rescheduling our flows. After migration all flows have to be rescheduled so the Platform® LSF scheduler is aware of the new SAS® executable. We started rescheduling from Management Console logged in as an administrator with a stored password. We soon realized that the flows were not being rescheduled. Instead, a new folder matching the administrator user name had been created in Flow Manager and flows were being added as new instances under this folder. This occurred because the Management Console had passed the cached administrator credentials over to the Platform® LSF scheduler software. To circumvent this problem we created an administrator profile that did not save the password. We were then prompted for credentials each time we rescheduled a flow. At that point we entered the credentials for SASLSFADMIN which was used originally to schedule the flow. Although it was less convenient, this enabled us to reschedule the flows instead of creating new ones.

User information is no longer transmitted with PROC OLAP.
Several of our cubes are refreshed automatically every night as part of a scheduled flow. After we redeployed these jobs according to the migration instructions, the scheduled jobs failed. The original PROC OLAP code included the user name and password of the account that created the deployed job. This is no longer the case. For security reasons, credentials are no longer included in code generated by Data Integration or OLAP Cube Studios. As a result, our newly deployed jobs failed because they could not access the metadata server. The jobs were being executed as SASLSFADMIN which was an operating system account created specifically for the Platform® LSF scheduler software. Since this account did not have an identity in metadata it was trying to access the cube as a member of the PUBLIC group which has been denied access to everything. To solve this problem, a SASLSFADMIN user was created in the Management Console. This user was then given Read, ReadMetadata, and WriteMetadata privileges in the permission pattern on all of our Access Control Templates.

Some MDX code stopped working.
We use conditional authorization on certain cube dimensions to prevent users from viewing detail data that they are not authorized to see. As with any type of programming, there is more than one way to construct the MDX expressions used to limit authorization. Some of our MDX expressions that had worked successfully in version 9.1 were not compatible with the new OLAP server. Web Report Studio returned an error page when users with this code tried to access a report built on their cubes. We were able to eliminate the error by modifying the code but were not particularly fond of the way the table was rendered in the Web Reports. There is a new option for cubes that allows you to “Include secured member values in presummarized computations.” The implementation of this new option and our original MDX code were interfering with each other. We were able to eliminate the error and get the best rendering of our table by deselecting this option and adding “[Unit].[All Unit],” to the beginning of our existing MDX code. Unit is the cube dimension where our MDX authorization conditions are applied.

THE WEB COMPONENTS ARE MUCH LARGER

Hopefully you will be able to migrate your BI environment to a destination where disk space is not an issue. It was not an issue for us in our production environment but it was in our development environment for our middle tier server. As mentioned earlier, resources for the virtual servers were limited so we tried not to ask for anything more than what we thought we absolutely had to have. We based our request on the size occupied by our initial installation. We requested 13.5 gigabytes to house all of the middle tier software for both versions. This would include all SAS middle tier software, the application servers, and the WebDAV data. Our test migration failed a number of times because we ran out of disk space and we were not able to free up enough space to complete the migration. The web components are larger in version 9.2 and a number of copies are made during the migration as application source files are exploded and deployed. The size of our Tomcat folder was around 1 GB. The corresponding JBoss folder takes up about 3 GB. The combined size of our BI configuration and Xythos folders was also around 1 GB. This data now takes up over 3 GB in the new BI configuration folder. We were able to complete the test migration by moving the JBoss application folder to our Windows system volume. This is the only aspect of our development environment that prevented it from being an exact replica of the production environment.

EXAMPLES DON'T ALWAYS WORK

Our internal security policies have always required us to encrypt the warehouse content using the HTTPS protocol. We use Internet Information Server (IIS) to do the encryption as an intermediary between the client browser and the BI application server. There are some additional configuration steps in IIS and JBoss to make this work. SAS technical support sent instructions describing these configuration steps. One of those steps involved uncommenting a couple of lines in a configuration file and adding parameters to the end of those lines. Instead of adding the parameters, we just copied the lines from the instructions and pasted them into the configuration file. After making this change we could not restart the JBoss service. After observing that the parameters in the instructions were in a different order than the parameters in the original configuration file we decided to go back to the original file and add the new parameter at the end of the lines. This fixed the problem. The order of the parameters in the configuration lines was critical and the order supplied in the instructions was incorrect. Unfortunately, there was a delay between the time we did this configuration and the time I put together this section of our internal migration script. Again I copied and pasted, forgetting that we were not able to use the literal text of the instructions. I wasted about an hour during the production migration trying to figure out why the JBoss service would not start. Fortunately, I had a working development environment with which to compare my configuration files.

Another of these configuration changes caused the system to be unresponsive. Closer examination of the settings revealed that they had set the logging level to "trace." This detailed level of logging took up so many of the web server's resources that it could not respond to session requests. The logging level was changed to "error" and the system was functional again.

It may not be practical or even possible to understand all of the configuration settings that are supplied by SAS. However, a certain amount of scrutiny is wise when implementing these changes. Always make a backup copy of the configuration file before changing it so you can easily undo your changes if they do not bring about the desired results.

THE FOLDER STRUCTURE MAY NOT MATCH DOCUMENTATION

The SAS documentation is based on an original installation of the BI Platform. The names of some of the physical folders under Lev1 are different in version 9.2. For example, the metadata server is now placed under SASMeta instead of SASMain. However, in a migration much of the old file structure is preserved. In our migration, we ended up with both a SASMain folder and a SASMeta folder. This can make it extra challenging to locate items such as logs and configuration files referenced in the BI manuals.

GET TO KNOW YOUR ROLES

Roles were originally introduced in SAS® Web Report Studio 2.1, but were managed by editing a configuration file, and changes required redeploying the web application. In SAS® 9.2, several applications, including Web Report Studio, use roles that are defined and managed in metadata to control features and functionality. Roles are now centrally managed in the metadata repository using SAS® Management Console. Essentially, the function of roles is to remove features and functionality from these client applications. We had not been using roles and did not intend to use them nor limit functionality. Therefore, they were not part of our research and testing for the migration project. It was not until I was working on some of our training documentation that I realized I could not find some of the functions in Web Report Studio. We discovered that all of our users had somehow been assigned a role that, by default, restricted the functionality of Web Report Studio. We were forced to learn enough about the roles to find a way to give the users back the functionality they needed. Even if you are not planning to use roles to place restrictions on your users, you should include this in your testing to ensure that user groups have all their intended capabilities.

YOU CAN'T CATCH EVERYTHING

It is difficult to imagine what users might do given the capabilities. This fact became obvious when we discovered that users were saving their custom reports in random metadata folders. This is when we became aware of the significance of the new browsing capabilities of Web Report Studio mentioned above. We tested access to existing content and we tested the capability of saving content where we intended for it to be saved but we did not consider the possibility of navigating outside these designated areas.

We also forgot about the process of creating a new scheduled job. It was almost three months after the migration when we needed to schedule a brand new job for the first time. This was a Data Integration Studio job that automatically generates the SAS® code. The job was deployed and scheduled then subsequently failed the first time the flow was executed. There is a default setting on the options tab of the job properties in Data Integration Studio that attempts to collect runtime statistics when running the job. This option requires ARM performance monitoring to be properly configured. This item escaped our notice and at this writing still has not been configured. For the time being we will disable the option to collect statistics when we are scheduling jobs.

You should expect be surprised but if you test thoroughly and learn from the mistakes of others such as ourselves, the frequency and severity of the surprises should be kept to a minimum.

CONCLUSION

The migration utility is very efficient as far as basic installation and configuration of the BI software is concerned. It did a good job of preserving our previous configuration and content. However, there is still a lot of work required before and after the migration. Much of that work involves preparation for and then adapting to the differences between the 9.1 and 9.2 versions especially when it comes to security. Hopefully, those who read this paper will have a head start and can avoid some of the pitfalls we experienced.

REFERENCES

SAS Institute Inc. 2009. *SAS® Intelligence Platform 9.1.3 to 9.2 Migration Guide, First Edition.* Cary, NC: SAS Institute Inc.

Wisniewski, Kathy 2010. "Be All That You Can Be: Best Practices in Using Roles to Control Functionality in SAS® 9.2," Proceedings of the 2010 SAS Global Forum, Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Faron Kincheloe
Baylor University
One Bear Place #97032
Waco, TX 76798
Phone: (254) 710-8835
Email: Faron_Kincheloe@baylor.edu

Appendix 1
Migration Work Flow

Step	Page	Who	Prerequisites	Complete/Notes
Pre-Migration Steps	1-2	JJM FDK	None	
Stop Tomcat	2	JJM	Friday 5pm	
Verify redirect URL points to down page	N/A	FDK	Friday 5pm	
Verify that all scheduled jobs ran successfully	N/A	JJM	Saturday am	
Migration Rainman & Topgun	2-7	All	Verification of Scheduled jobs	
Client Installations	8	JJM	Migrations	
Manual Configuration of JBoss & IIS for HTTPS	9-20	FDK	Migrations	
Management Console Configuration for HTTPS	21-25	FDK JJM	Configuration of JBoss & IIS	
Security Changes	26-30	JJM	Client Installations	
SAS Add-ons, Patches & Manual Configuration	31-32	FDK	Migrations	
Update SAS Programs for WebDAV, etc.	33	SSH	Migrations	
Rebuild OLAP Cubes	34	SSH	Client Installations	
Update Scheduled Jobs	35	JJM FDK	Security Changes	
Update Metadata for HR & Fin. Tables	36	FDK SSH	Update Scheduled Jobs & Update SAS Programs	
Reload Cube MDX	37	JJM SSH	Rebuild Cubes	
Update Portal Pages & Timeout Settings (Can be split up)	38-41	JJM FDK	Client Installations & Manual Configuration of JBoss & IIS	
Reorganize BI Folders	42	SSH	Management Console Config for HTTPS (Portal & WRS must be up)	
Test Group Access & MDX security	N/A	All	All the above	
Send new URL for redirect to Web Administrator	N/A	FDK	All the above	
Send message to Bearhaus-L	N/A	SSH	All the above	
Send DOWN line clear to HelpDesk	N/A	FDK	All the above	